```
**********************************************************
   40638 Mon Feb 15 17:29:58 2016
new/usr/src/uts/common/fs/ufs/ufs_subr.c
6660 ufs may read too many indirect blocks, flush a random block to disk
**********************************************************
_____unchanged_portion_omitted_
 565 /*
 566  * Flush all indirect blocks related to an inode.
 567  * Supports triple indirect blocks also.
 568  */
 569 int
 570 ufs_sync_indir(struct inode *ip)
 571 {
 572         int i;
 573         daddr_t blkno;
 574         daddr_t lbn;    /* logical blkno of last blk in file */
 575         daddr_t clbn;   /* current logical blk */
 576         daddr32_t *bap;
 577         struct fs *fs;
 578         struct buf *bp;
 579         int bsize;
 580         struct ufsvfs *ufsvfsp;
 581         int j;
 582         daddr_t indirect_blkno;
 583         daddr32_t *indirect_bap;
 584         struct buf *indirect_bp;

 586         ufsvfsp = ip->i_ufsvfs;
 587         /*
 588          * unnecessary when logging; allocation blocks are kept up-to-date
 589          */
 590         if (TRANS_ISTRANS(ufsvfsp))
 591                 return (0);

 593         fs = ufsvfsp->vfs_fs;
 594         bsize = fs->fs_bsize;
 595         lbn = (daddr_t)lblkno(fs, ip->i_size - 1);
 596         if (lbn < NDADDR)
 597                 return (0);     /* No indirect blocks used */
 598         if (lbn < NDADDR + NINDIR(fs)) {
 599                 /* File has one indirect block. */
 600                 blkflush(ip->i_dev, (daddr_t)fsbtodb(fs, ip->i_ib[0]));
 601                 return (0);
 602         }

 604         /* Write out all the first level indirect blocks */
 605         for (i = 0; i < NIADDR; i++) {
 605         for (i = 0; i <= NIADDR; i++) {
 606                 if ((blkno = ip->i_ib[i]) == 0)
 607                         continue;
 608                 blkflush(ip->i_dev, (daddr_t)fsbtodb(fs, blkno));
 609         }
 610         /* Write out second level of indirect blocks */
 611         if ((blkno = ip->i_ib[1]) == 0)
 612                 return (0);
 613         bp = UFS_BREAD(ufsvfsp, ip->i_dev, (daddr_t)fsbtodb(fs, blkno), bsize);
 614         if (bp->b_flags & B_ERROR) {
 615                 brelse(bp);
 616                 return (EIO);
 617         }
 618         bap = bp->b_un.b_daddr;
 619         clbn = NDADDR + NINDIR(fs);
 620         for (i = 0; i < NINDIR(fs); i++) {
 621                 if (clbn > lbn)
 622                         break;
 623                 clbn += NINDIR(fs);
```



```
 624                 if ((blkno = bap[i]) == 0)
 625                         continue;
 626                 blkflush(ip->i_dev, (daddr_t)fsbtodb(fs, blkno));
 627         }

 629         brelse(bp);
 630         /* write out third level indirect blocks */

 632         if ((blkno = ip->i_ib[2]) == 0)
 633                 return (0);

 635         bp = UFS_BREAD(ufsvfsp, ip->i_dev, (daddr_t)fsbtodb(fs, blkno), bsize);
 636         if (bp->b_flags & B_ERROR) {
 637                 brelse(bp);
 638                 return (EIO);
 639         }
 640         bap = bp->b_un.b_daddr;
 641         clbn = NDADDR + NINDIR(fs) + (NINDIR(fs) * NINDIR(fs));

 643         for (i = 0; i < NINDIR(fs); i++) {
 644                 if (clbn > lbn)
 645                         break;
 646                 if ((indirect_blkno = bap[i]) == 0)
 647                         continue;
 648                 blkflush(ip->i_dev, (daddr_t)fsbtodb(fs, indirect_blkno));
 649                 indirect_bp = UFS_BREAD(ufsvfsp, ip->i_dev,
 650                     (daddr_t)fsbtodb(fs, indirect_blkno), bsize);
 651                 if (indirect_bp->b_flags & B_ERROR) {
 652                         brelse(indirect_bp);
 653                         brelse(bp);
 654                         return (EIO);
 655                 }
 656                 indirect_bap = indirect_bp->b_un.b_daddr;
 657                 for (j = 0; j < NINDIR(fs); j++) {
 658                         if (clbn > lbn)
 659                                 break;
 660                         clbn += NINDIR(fs);
 661                         if ((blkno = indirect_bap[j]) == 0)
 662                                 continue;
 663                         blkflush(ip->i_dev, (daddr_t)fsbtodb(fs, blkno));
 664                 }
 665                 brelse(indirect_bp);
 666         }
 667         brelse(bp);

 669         return (0);
 670 }
_____unchanged_portion_omitted_
```